Security for SAS® Servers
Brian Perkinson, SAS Institute Inc., Cary, NC

ABSTRACT

Controlling access to services and data is an important consideration in client/server computing. Interposed between client SAS sessions and the data they access, the multi-user server provided by SAS/SHARE® software must preserve the existing protections on that data.

This paper is a thorough treatment of security in the multi-user SAS server. It covers the issues and challenges presented by a distributed, heterogeneous environment. It describes the controls available to an application developer or server administrator for controlling access to the server and the data available through it. This paper also discusses the host mechanisms used by the server to authenticate a user's identity and check his or her authority to access SAS libraries and files.

INTRODUCTION

Controlling access to data is a relatively manageable problem when a computer system is closed and isolated. A single personal computer in a lab or office, unconnected to any network, can be secured by physical means, by limiting who can walk up to it and use it. Even a multi-user system like a mainframe or minicomputer can be secured relatively easily when not part of a network; administration and management of the system typically place the computer and its operator consoles in restricted-access areas and users are required to log in so that the system itself can verify and record their actions. In such a single-system environment, the responsibility for validating a user's identity and his or her authority to access resources is placed at very low level, usually in the operating system itself.

The shift to distributed processing and client/server computing, however, opens up the system and significantly increases the potential for problems. It also presents special challenges for insuring the security of the networked resources. The computers on the network are typically distributed physically, sometimes geographically, and end user machines frequently require no log-in process. Data becomes more widely available and is not read and written via physical I/O operations but by communicating with an independently-running program. Access control can no longer reside in the physical I/O layer of the operating system because all physical I/O is performed by the server on behalf of the true end users. Nor can it be handled by the communications code, which carries messages between user and server but isn't aware of their content. In a client/server computing model, the responsibility for controlling access to data and services must be placed higher, in a layer that can distinguish the identity and actions of various end users.

DEFINITIONS

The general concept of security encompasses two separate but related concerns: authentication and authority. Authentication is the process of verifying the identity of a user of a system or program. Authority is the permission to access a resource, granted by its owner or administrator. Authority may be determined by verifying a password associated with the resource and supplied by the user or by checking a security profile or access control list that describes which users and groups of users may access the resource and how. This latter, identity-based authority checking depends on authentication: such a check is meaningless unless the identity has been established conclusively.

ACCESS CONTROLS IN SAS SERVERS

The multi-user SAS server offers several mechanisms for controlling access to its resources: server access passwords, administrator control of what SAS libraries can be defined to the server, passwords on SAS data sets, views, and catalogs, user authentication, and identity-based host file authority checking. Each mechanism has advantages and disadvantages; developers and administrators can use them in combination to address different needs and requirements.

SERVER ACCESS PASSWORDS

The simplest way to control who can access data through a server is to control who can access the server itself. A server access password also offers the broadest control over the server and its resources: it applies to all users regardless of what data they access and what operations they perform.

A server administrator can use this password to restrict access to data without having to protect the data itself. For example, a server used by a human resources department might permit access to personnel and payroll data. By defining a server access password known only to users in the human resources department and to the applications they use, the administrator can prevent users in other departments from accessing this sensitive data through the server.

If used by itself, the server access password does have its disadvantages. It offers broad control but no specific control; either a user can connect to the server and access all its data or he cannot. And it only prevents access to a server's data through that server, offering no protection against a user accessing that data through
another server or directly via physical I/O. But used with other mechanisms it is a powerful line of defense.

A server administrator defines a server access password for users via the UAPW= option on the PROC SERVER statement. (There is also a server access password for administrators, who use PROC OPERATE, defined with the OAPW= option.) Users specify the password via the SAPW= option on the LIBNAME statement and the PROC SQL CONNECT TO statement.

LIMITED SERVER ACCESS

A server administrator can also control access to data through the server by limiting what data that server can access. This control can take two forms: he or she can prevent users from defining libraries to the server or use host file security to permit the server access only to certain SAS libraries and files.

An administrator can prevent users from defining libraries to the server by specifying the NOALLOC option on the PROC SERVER statement. This confines users to accessing only those libraries defined to the server by the administrator. The NOALLOC option effectively guards against undesired access to data that is not defined to that server.

The administrator can also prevent users from accessing certain data through the server using host file security to deny the server itself access to that data. The server is subject to the control enforced by the operating system or other host file security software just as any other program is. If the server is only authorized to access one library or just a subset of the members of a library, its users will only be able to access that library or those members through that server.

Used alone, each of these mechanisms permits an administrator to define a set of data that is accessible to anyone who can connect to the server. For example, an employee database application might offer non-sensitive data such as room number, phone extension, or department name. By preventing users from defining other libraries to the server or preventing the server from accessing other libraries, an administrator could eliminate concerns over who could access the server.

These mechanisms can also be used in combination with server access passwords to define departmental servers. For example, a server access password would prevent users outside an accounting department from accessing that department's server; the NOALLOC option or host security limitations would prevent the accounting department from accessing other data through its server.

SAS FILE PASSWORDS

The SAS System permits a user to define passwords that must be supplied in order to access a SAS data set, view, or catalog in a particular way. Though not limited to use with SAS/SHARE software, these passwords take on added importance in the context of the multi-user server. The SAS server makes a community of users out of a collection of formerly separate and independent individuals and groups. An application developer or administrator can assign passwords that must be supplied in order to read, write, or alter the characteristics of a SAS file. These passwords help prevent unauthorized access to data directly and through a server.

SAS file passwords permit much finer control over access to data than server access passwords and limited access to the data by the server. By the same token, they impose a greater administrative burden, since there can be many different passwords specified many different places throughout an application.

HOST-BASED MECHANISMS

Most of the mechanisms described thus far are implemented entirely within the SAS System and thus are portable. An administrator who relies on server access passwords, the PROC SERVER NOALLOC option, and SAS file passwords can move the data and the server it is accessed through to a different host platform and all of these controls remain intact.

By contrast, user authentication and identity-based authority checking depend on interaction with the host security software. The advantage of these mechanisms is that the infrastructure of userids and file permissions, security profiles, or access control lists is already in place when a customer introduces SAS/SHARE software. Furthermore, administrators are already familiar with these mechanisms and how to use them, which translates into greater maintainability and ease of use. However, the client/server model presents significant challenges to supporting these mechanisms.

CLIENT/SERVER CHALLENGES

We have said that the client/server model requires that authority checking be handled by a code layer high enough to know what resource the user wants to access and what operation they want to perform on it. In the context of a single, monolithic system, an identity-based authority check can be made in the client. In a distributed environment, however, this kind of authority check must be made in the server because the server necessarily runs on the system where the data resides and the software that controls access to it executes. Furthermore, the server environment is a more secure environment, typically under the control of a trusted administrator.
A heterogeneous network presents a further requirement for identity-based authority checking. It isn't enough that the client user's identity is verified; that identity must be known to the authority-checking software the server uses. Thus the server must authenticate a client user in terms of its own security software and then must use that authenticated identity to verify the user's authority to access services and data.

AUTHORITY CHECKING

Identity-based host file authority checking relies on the host file system or separate security software installed on the host to keep track of who has what kind of access to a file. When a client user wants to assign a SAS library or open a SAS file in a certain way, the SAS System either asks the host file system or security software whether the user has the authority to do so or it asks for the permissions associated with the file and decides for itself. In either case, the decision is made based on who the user is and what the host file system or security software has been told about who may access the file and how.

For example, suppose a SAS data set protected by conventional Unix file permissions may be read and written by its owner, read by other users in the owner's user group, and may not be accessed at all by users in other groups. When the owner attempts to open the data set for output, the SAS System obtains the file permissions from the operating system, determines that the user is the owner, and permits the open. When a user from another group tries to open the data set for output, the SAS System obtains the permissions, determines that the user is not the owner and not in the owner's group and thus has no access to the file, and rejects the open.

As long as the user in this example is running on the Unix system where the SAS data set resides, this checking could be done in the user's SAS session. However, SAS software supports communication between a server and user running on different host platforms. The client user in our last example may be accessing the Unix data set through a Unix server from a SAS session on a PC running Windows. In such a situation the remote engine clearly has no way to verify the user's authority to access a SAS library or file. So the authority check is performed in the server.

However, the part of the server that knows when and how a user wants to access a SAS library or file is portable. It must use routines provided in the host layer of the SAS System to make a host security call to check a user's authority to access SAS libraries and files. The implementation of these routines necessarily varies from one host platform to another.

USER AUTHENTICATION

To make an identity-based host file authority check, the server requires an authenticated userid known to the host file system or security software on the host where the server runs. Some of the access methods used for communication between a server and a client user session require that both sessions be running on the same system. Such access methods can thus present the userid of the client session to the server without having to perform any authentication.

Other communications access methods permit inter-host connectivity. If an authenticated userid is required for a connecting user, such an access method must perform the authentication by requiring the user to supply a valid userid and password for the server's host system.

An access method that presents authenticated userids to the server is said to be running secured. Access methods that provide inter-host connectivity do not run secured by default. The server administrator must set a SAS system option, SAS macro variable, or host environment variable for each access method used in the server's session that is to run secured. For example, to run the TCP/IP access method secured in a server session on a UNIX workstation, the administrator must set an environment variable or macro variable called TCPSEC to _SECURE_.

An access method that does not present authenticated userids to the server is said to be running unsecured. When an access method is running unsecured, the server bypasses all host file authority checks for users who communicate via that access method. It is possible to run some access methods secured and others unsecured in the same server session.

By default, servers on MVS and CMS require that all access methods run secured while servers on other hosts permit access methods to run unsecured. On these latter hosts, a server can be made to require all access methods to run secured by specifying the options UVALID=YES and OVALID=YES on the PROC SERVER statement. On MVS and CMS, servers can be made to permit one or more access methods to run unsecured by specifying either UVALID=NO or OVALID=NO. The UVALID= and OVALID= options specify whether an authenticated userid is required from connecting users and administrators, respectively. It is important to note that even if UVALID=NO is specified, a connecting user will still have to supply a valid userid and password if the access method he or she uses to communicate is running secured in the server session.

In addition to host file authority checking, running an access method secured provides an additional measure of access control by requiring users to have a valid userid for the server's host system in order to connect to it. This means that any SAS library or file that is universally accessible as far as the host file system or security software is concerned is still only available to users who
could actually logon to that system.

For example, an administrator for a server on MVS might wish to make all the SAS libraries and files that are to be accessed through that server available to anyone who could logon to MVS. However, he might not wish to make them available to the larger community of PC and workstation users who do not have MVS users. He could implement that restriction by leaving the SAS libraries and files unprotected or world-readable but running the inter-host access methods secured.

HOST IMPLEMENTATIONS

MVS

Authentication and authority checking are performed on MVS by calling the MVS system authorization facility (SAF). These calls are handled by the SAF router. The router forwards the calls to the security software installed on the system, which may be IBM's RACF or a third-party product such as Computer Associates' CA-ACF2.

Three access methods provide inter-host connectivity on MVS: VTAM LU 6.2, APPC, VTAM LU 0, and TCP/IP. The VTAM LU 6.2 and TCP/IP access methods authenticate a user by issuing a RACROUTE REQUEST=VERIFY. This call validates the specified userid and password and must be made by an authorized program. For this reason the access methods actually call the SAS supervisor call (SVC) routine to issue the RACROUTE.

The VTAM LU 0 access method has been replaced by the VTAM LU 6.2 access method and is provided only for compatibility with previous releases of SAS/SHARE software. The VTAM LU 0 access method does not run secured but will run trusted; in this mode, the client userid is not authenticated but is presented to the server as if it were. This causes the server to make authority checks for users who connect via the VTAM LU 0 access method. Trusted mode is provided as an option for server administrators who know that the userids from the client session will be meaningful in the server's session.

The host layer routine called by the server to check a user's authority to access a SAS library issues a RACROUTE REQUEST=AUTH. This call also requires program authorization and is also issued from the SAS SVC.

The security software knows nothing about the individual members of a SAS library. Therefore it can only be used to control access to the library as a whole. For this reason, the host layer routine for checking a user's authority to access a SAS library member always returns success on MVS.

It is worth noting that some security software packages, including older releases of CA-ACF2, do not support third-party authentication. Also, some security products, including CA-ACF2, do not by default make real authority checks in response to RACROUTE calls. Instead they always return success unless SAF interface support is enabled in the security software by the site security administrator.

CMS

Authentication and authority checking are handled on CMS by customer-installed exit routines. Working sample versions of these routines are supplied by the Institute for the major security software products for CMS, IBM's RACF and Sterling Software's VM:Secure. Customers with other third-party security software or with home-grown systems can create their own routines to be used in place of these samples.

Two communications access methods provide inter-host connectivity on CMS: APPC and TCP/IP. For the APPC access method, authentication is performed by APPC itself. The TCP/IP access method authenticates incoming users by calling the customer-installed exit routine.

The host layer routines called by the server to check a user's authority to access SAS libraries and library members in turn call the customer-installed exit routines. These routines receive as parameters the authenticated userid, the file type and mode (for a library) or full file id (for a library member), and a flag indicating whether the user wants read or write access.


OpenVMS

Support for authentication and authority checking is available in Release 6.0BE of the SAS System for OpenVMS on the VAX and Release 6.11 of the SAS System for OpenVMS on the Alpha AXP. The DECnet and TCP/IP communications access methods call the OpenVMS system routine SYSS$GETUIU to validate the userid and password for connecting users. This routine requires that the server process have the BYPASS or SYSPRV privilege.

The host layer routines called by the server to check a user's authority to access SAS libraries and files call the OpenVMS system routine SYSS$CHECK_ACCESS to obtain the privileges associated with the specified directory or file. The host layer routines compare the UID for the authenticated userid to these privileges to determine if the client user has the requisite authority. No special privileges are required to call SYSS$CHECK_ACCESS.
Systems Architecture

Windows NT

Support for authentication and authority checking is available in Release 6.11 of the SAS System for Windows NT. The TCP/IP access method authenticates connecting users via the system routine LogonUser. This routine requires version 3.51 of Windows NT. To call this routine the server process must have "act as part of the operating system" privilege.

Authority checking for authenticated users is handled by calling the system routine AccessCheck. This routine requires a token created by LogonUser and thus also requires version 3.51 of Windows NT.

Unix

Support for authentication and authority checking is available in Release 6.11 of the SAS System for Unix. The TCP/IP access method authenticates connecting users by calling a customer-installed external program called authenticate. This program takes pointers to a userid and a password and returns success or an error code. Typically it validates the userid by encrypting the password and comparing it to the encrypted password for the specified userid in /etc/passwd. Working sample versions of this program are provided for standard Unix systems and for those that use shadow password files (in which the encrypted passwords are not stored in the world-readable /etc/passwd but in a file that can only be read by a program running as root). The shadow password version of this program must be installed and run as root.

The host layer routines called by the server to check a user's authority to access a SAS library or file call a customer-installed external program called permission. This program takes pointers to the authenticated userid, the pathname of the directory or file, and a flag indicating whether the user wants read or write access. This program typically calls stat(2) to obtain the file permissions and uses them to validate the specified user's authority to the specified file or directory. Working sample versions of this program are supplied for ordinary file permission checking, AFS ACLs, and AIX ACLs. On some versions of Unix this program must be installed and run as root.

For full details on these customer-installed external programs, refer to SAS/SHARE Technical Report for Unix Environments, Release 6.11 and to the README files in the utilities/src subdirectory of the sasroot directory.

FUTURE DIRECTIONS

As heterogeneous networks become more prevalent, the inherent security challenges become more important to those who administer those networks and to those who use them. Mechanisms such as third-party authentication and single signon promise to ease the end user burden of supplying userids and passwords for multiple systems over and over as connections are made and dropped. In Version 7 of the SAS System we hope to address this problem by supporting these and other mechanisms while continuing to support host-based authentication and file security.

SUMMARY

The shift to client/server computing over a distributed, heterogeneous network presents significant challenges for controlling access to data and its processing. The SAS System meets those challenges by enabling a server administrator to control who can access the server and what libraries are accessible through it, by providing passwords for various kinds of access to SAS files, and by authenticating client sessions and enforcing the restrictions imposed by host file security on the users they represent. As other mechanisms develop and mature, the SAS System will be extended to support them in order to continue to meet the needs and requirements of customers.

REFERENCES


SAS and SAS/SHARE are registered trademarks of SAS Institute Inc. in the USA and other countries. IBM, RACF, and VTAM are registered trademarks or trademarks of International Business Machines Corporation. © indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

Brian Perkinon
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
(919) 677-8000 x7554
sasbrp@sas.com